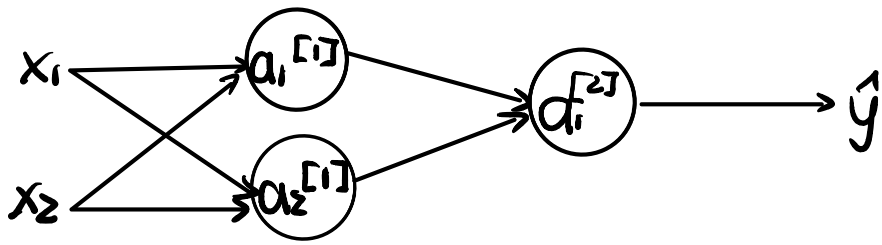
Random Initialization

What happens if weights are not initialize randomly?

Example with weights all initialized to zero:



$$W^{[1]} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad b^{[1]} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$W^{[2]} = \begin{bmatrix} 0 & 0 \end{bmatrix} \qquad b^{[2]} = \begin{bmatrix} 0 \end{bmatrix}$$

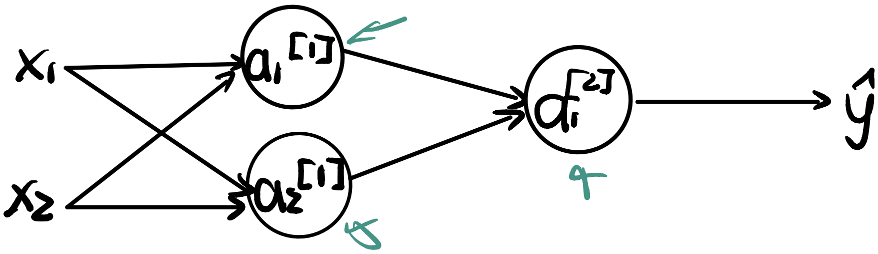
What would happen during forward calculation:

$$a_1^{[1]} = a_2^{[1]} \qquad dz_1^{[1]} = dz_2^{[1]}$$

$$\Rightarrow \quad dw = \begin{bmatrix} u & v \\ u & v \end{bmatrix}$$

All hidden layer neurons will produce same results and have same gradient during backward propagation.

Random Initialization:



$\rightarrow w^{[1]}$ = np.random.randn((2,2)) * 0.01

$b^{[1]}$ = np.zeros((2,1))

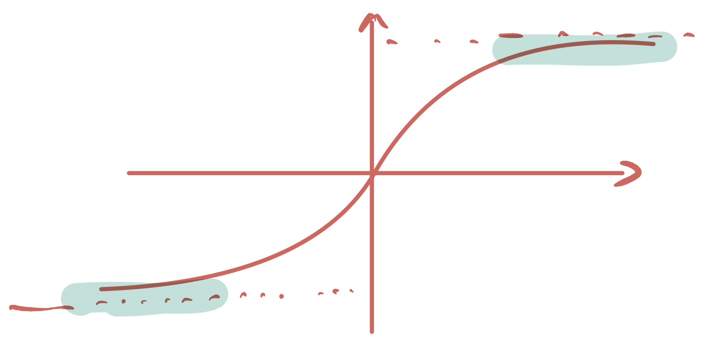
$w^{[2]}$ = np.random.randn((1,2)) * 0.01

$b^{[2]} = 0$

why do we need to scale them down?

Initialize biases to zeros won't cause the symmetric problem or slow down learning.

Normally, they are set to zeros by default.



higher input to tanh, sigmoid functions, cause gradient to be close to zero, making the learning/optimization slow.